UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/767,365	01/22/2001	Sheng Liang	2006579-0558 (CTX-199)	2538

69665 7590 10/27/2009

CHOATE, HALL & STEWART / CITRIX SYSTEMS, INC.
TWO INTERNATIONAL PLACE
BOSTON, MA 02110

EXAMINER	
TRAN, QUOC A	

ART UNIT	PAPER NUMBER
2176	

MAIL DATE	DELIVERY MODE
10/27/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	09/767,365	LIANG ET AL.	
	Examiner	Art Unit	
	Quoc A. Tran	2176	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 August 2009*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,4-6,8,11-13 and 15-18* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,4-6,8,11-13 and 15-18* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *26 August 2009* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

This is Non-Final Office Action in response to the RCE/amendments/remarks filed 08/26/2009. The current patent application originally filed 01/22/2001, which claims benefit to provisional No.60/228,904 filed ***08/29/2000***,

- Claims 1, 4-5, 6, 8, 11-12, 13, and 15-18 are pending.
- Claims 1, 6, 8, 11, 13, 15 and 16 are independent claims.
- Claims 2-3, 7, 9, 10, and 14 are canceled.
- Claims 16-18 are new.

It is notes the objection to the specification set forth in the previous office action dated 05/26/2009 is hereby withdrawn.

Examiner's Comments Concerning Claim 15 Objection to the Specification,

Claim 15 recites an "*intermediary*" for efficiently parsing receive data transmitted between a client and a server (e.g., see Claim 15 the preamble).

For examination purposes, the examiner interprets the recited "*intermediary*" to necessarily include DSP item 100 of figure 1 (e.g., Derivative Service Provider), as indicated in the specification at Page 5 lines 13-20 and illustrated in figure 1 item 100. Also as indicated in the applicant's remarks filed 08/26/2009 @ Page 8 second half though Page 9 first paragraph.

Continued Examination Under 37 CFR 1.114

A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 08/26/2009 has been entered.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 13 and 15 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claim 13:

In summary, Claim 13 recites a "*system*" comprising a "*virtual browser*," an "*identification engine*," a "*cache*," a "*comparison engine*" and a "*parsing engine*." Although the Specification of the present application fails to expressly indicate whether the recited elements are computer **hardware** components or computer **software** component, one of ordinary skill in the art would interpret them as being software. Likewise, for purposes of examination, the examiner will interpret each of these elements as computer **software** elements. Thus, the recited "*system*" is software per se.

Accordingly, the "*system*" is not a "process," a "machine," a "manufacture" or a "composition of matter," and Claim 13 fails to recite statutory subject matter, as defined in 35 U.S.C. 101.

Claim 15:

In summary, Claim 15 recites a "*intermediary*" comprising a "*cache*," a "*comparison engine*" and a "*virtual browser*." Although the Specification of the present application fails to expressly indicate whether the recited elements are computer **hardware** components or computer **software** component, one of ordinary skill in the art would interpret them as being software. Likewise, for purposes of examination, the examiner will interpret each of these elements as computer **software** elements. Thus, the recited "*intermediary*" is software per se.

Accordingly, the "*intermediary*" is not a "process," a "machine," a "manufacture" or a "composition of matter," and Claim 15 fails to recite statutory subject matter, as defined in 35 U.S.C. 101.

Claims Rejection – 35 U.S.C. 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> ***(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which***

said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1, 4-6, 8, 11-13 and 15 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Krishnamurthy et al.,** (US 20060242145A1- Division of 10/177,783- filed 04/04/2002) [hereinafter "Krishnamurthy"], in view of **Jamtgaard et al.,** (US006430624B1 - filed 02/14/2000) [hereinafter "Jamtgaard"].

Regarding ***independent claim 1***, Krishnamurthy teaches:

> **a computer-implemented method for efficiently parsing received data file, comprising: receiving a data file; retrieving a stored version of the data file and a syntax tree comprising nodes and tokens representing data within the data file, the tree include at least one static node.**

(See figure(s) 4-5 and at page 5 Paragraph [0081] →Krishnamurthy teaches this limitation, as clearly indicated in the cited text [e.g., improving TreeDiff performances by: reading and parsing a web page (e.g. a data file) into an abstract syntax tree (AST). The syntax tree is then linearized into a sequence of tokens, which consist of markup elements (defined by the markup language syntax and denoting structure, semantics, formatting or other information) and text strings that represent the content. Also Krisnamurhty further discloses the syntax tree includes the distinguished node(s) from the old page (e.g., static node) and newpage wherein the old page's is preserved in some form inside the new page (dynamic page), and if so, the subtree is formed (html page), see Krisnamurhty at para [0129] and illustrates in figure 7).

comparing by comparison engine, the stored version of the data file with the received data file to identify non-matching content in the received data file,

(See Figure(s). 5 and 7 and page 5 Para [0077] --> Page 6 Para [0083]→ Krisnamurhty discloses this limitation that is defining a different between HTML pages by comparison utilized HTML*Diff* algorithm and compare the parse tree or abstract syntax tree representations of the documents, using subtree equality (or some weaker measure) as a basis for comparison. In this case, a subtree representing nodes and tokens to one of the subtrees and identify only differences of the HTML to output a subtree (figure 7).)

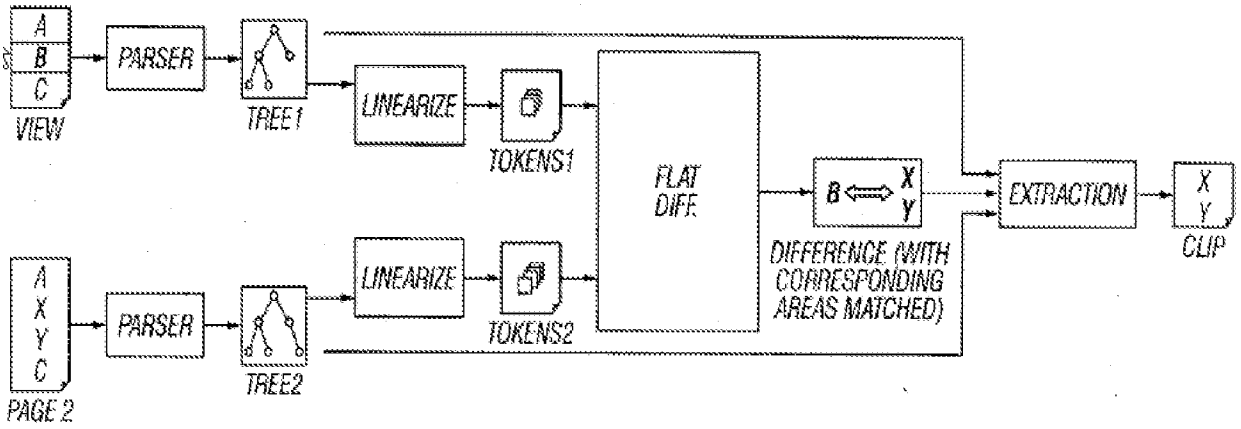

FIG. 5

parsing by a parsing engine, only the non-matching content of the received data file to form at least one subtree comprising nodes and tokens representing the non-matching content of the received data file;

(See Figure. 7 and 8 and @ Page 5 Para [0076] --> Krisnamurhty discloses **tracking** changes in pages by **computing page differences** (e.g., computing **the non-matching content**); that is only transmitting **the page difference** (non-matching content) to **reconstruct the new page**. Also Krisnamurhty further discloses to compute the edit sequence between two subtrees (or trees) [which means to compute **the differences** between two documents of two trees [@ Page 5 Para [0067]); first linearized the two subtrees into two token sequences; then perform on these two token sequences **2-way FlatDiff**: to identify the sub-token sequence by locating the boundary point in each direction, then feed this pruned sub-subtree into the vanilla TreeDiff in place of the un-pruned subtree. We combine the result of the vanilla TreeDiff with the result of the FlatDiff to form the final answer. As a result of this optimization, the size of the subtrees participating in most invocations of the vanilla TreeDiff method is significantly smaller [@ figure 8 and page 9 Para [0128]]

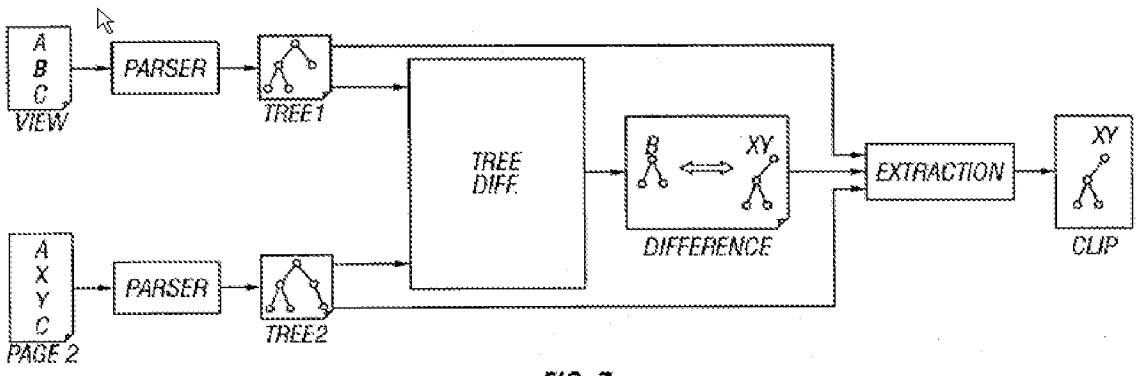

FIG. 7

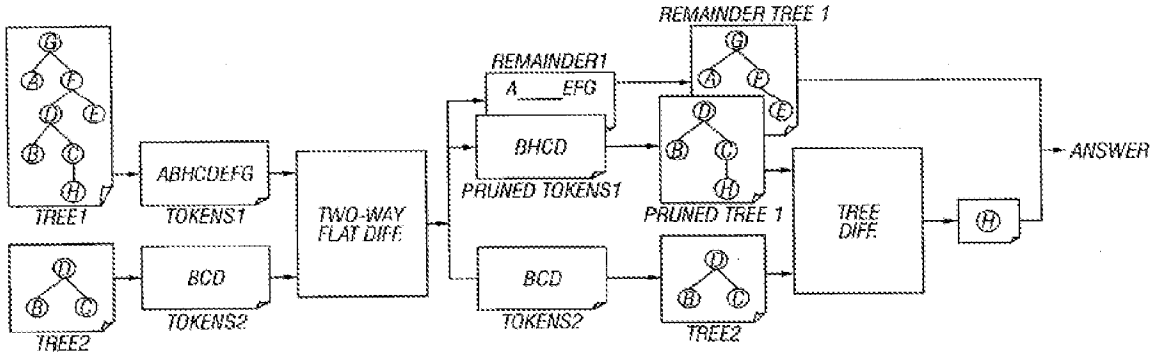

FIG. 8

This interpretation is supported by the applicant's disclosure, which is stated, “System 500 … newly retrieved version of the page and the cached copy of the page are compared using a binary **"diff" algorithm**, which identifies the **differences between the binary representation of two documents**"(See current disclosure page 11 lines 11→Line 25).

replacing, at least one static node of the syntax tree with a token; and creating, a mapping from each token to one of the subtrees.

(See Figure. 5 and page 5 Para [0077] --> Page 6 Para [0083], and Para [0129]→ Krisnamurhty discloses the Improving TreeDiff Performance: Subtree matching wherein the difference algorithms compute the mapping from the nodes in T1 to the nodes in T2. Given such a mapping, we can identify whether the distinguished node(s) from the old page (static) is preserved in some form inside the new page (dynamic), and if so, the subtree is formed.)

In addition, Krishnamurthy does not expressly teach, but Jamtgaard teaches:

A virtual browser,

(@ Figure 7 and @ Col. 10 Lines 21-47→ Jamtgaard discloses this limitation, as clearly indicated in the cited text [e.g., the content 20 is fed into ***a virtual browser*** 100 as described briefly above which passes on the content to the XML engine 46. Using the particular page-type rules sets stored in the database 47, the XML engine 46 may convert the content into relational markup language (RML) format 24. In more detail, a session on the Internet is handled by the virtual browser 100 located on the translation server 12. This virtual browser provides the important functionality of proxying javascript (*using a Javascript proxy engine 102*) and cookies (using a cookie proxy engine 104) for the target devices.)

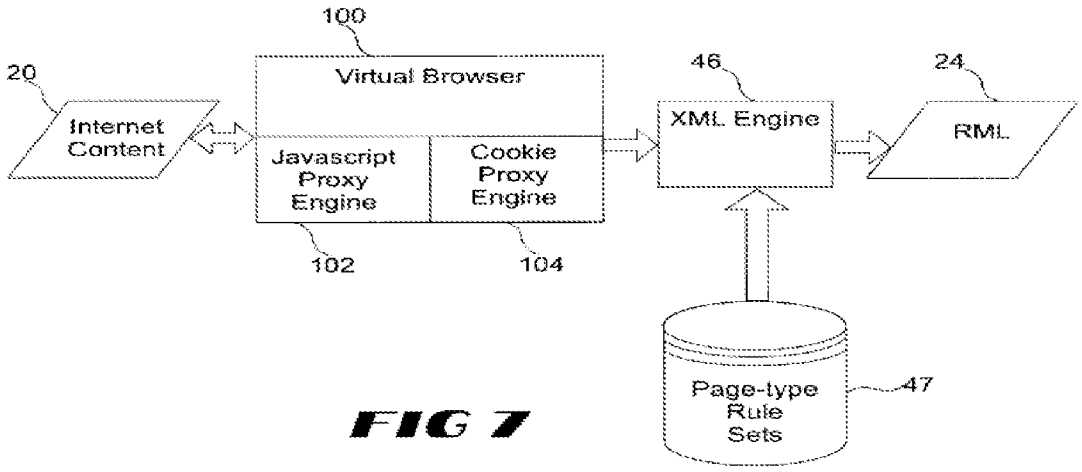

FIG 7

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified Krisnamurhty's method for efficiently parsing received data file that is related to the HTML *Diff* algorithm, and utilizing a proxy and some graphical user interface (GUI) code and sending the augmented page (named

page 1') to the user (named view client) (see Krisnamurhty at page 4 para[0063], to include the virtual browser as taught by Jamtgaard, because they are from the same field of endeavor of Viewing and parsing , and provides a predictable result of said content delivery may include intelligently harvesting content from a web page to provide multiple different web servers to send information to various information appliances. In particular, the system receives incoming content on-the-fly from an Internet content provider thereby allowing for dynamic information generation utilized the virtual browser; see Jamtgaard @ at column 4 lines 21-26 & @ Col. 10 Lines 21-47.

Claim 4,

Krisnamurhty and Jamtgaard teach the method of claim 1 and further comprise:

wherein the data file is a web page,

(See Paragraph [006]→ Krisnamurhty discloses web page is data file.)

Claim 5,

Krisnamurhty and Jamtgaard teach the method of claim 1 and further comprise

wherein the data file is an HTML file,

(See Paragraph [006 and 0052]→ Krisnamurhty discloses data file is HTML file.)

Regarding ***independent claim 6,***

Claim 6 is fully incorporated similar subject of claims 1 and 4-5 cited above, and are similarly rejected along the same rationale. Thus, Krishnamurthy and Jamtgaard

disclose every limitation of Claim 6 and provide proper reasons to combine, as indicated in the above rejections for Claims 1 and 4-5.

Regarding ***independent claim 8,***

Claim 8 are fully incorporated similar subject of claims 1, and 4-5 cited above, and are similarly rejected along the same rationale. Thus, Krishnamurthy and Jamtgaard disclose every limitation of Claim 8 and provide proper reasons to combine, as indicated in the above rejections for Claims 1, 4-5.

In addition, Krishnamurthy teaches:

a first HTML page; responsive to a determination that a cached version of the HTML page exists:

(See figure(s) 4-5 and at page 5 Paragraph [0081] →Krishnamurthy teaches this limitation, as clearly indicated in the cited text [e.g., reads and parses a web page (HTML page) from a cache into an abstract syntax tree (AST).)

responsive to a determination that the cached version of the HTML page exist: parsing, by the parsing engine, the received HTML page to form a second syntax tree comprising nodes and tokens representing the non-matching content of the received data file, the second tree containing at least one static node; and storing the second tree and the received HTML page in the cache.

(Krishnamurthy teaches retrieving a stored version of a web page and a syntax tree comprising nodes and tokens representing the data file (See figure(s) 4-5 and at page 5 Paragraph [0081] and [0129], e.g., to improving TreeDiff performances by: reading and parsing a web page (e.g. a data file) into an abstract syntax tree (AST)). After the web page is parsed into the form of an abstract syntax tree (AST); retrieving the stored version of the data file with the received data file in the form of AST is being compared to identify non-matching content (new/dynamic content) using the TREEDIFF algorithm, (see Figure. 5 and page 5 Para [0077] --> Page 6 Para [0083], e.g., TreeDiff algorithm). In addition, Krishnamurthy is parsing only the TREEDIFF (the non-matching content between the VIEW (A, B, C) and PAGE2 (A, X, Y, C)) to form a subtree of the non-matching content. This is generally set forth at page 8 Para [0107-0108] and illustrated in figure 7 of Krishnamurthy. Also see Krishnamurthy at para [0161], discloses the recent pages and their recently extracted clips are stored in or near the machine that performs extraction enables them to be reused like a cache.)

Regarding ***independent claim 11,***

Claim 11 is fully incorporated similar subject of claim 1 cited above, and is similarly rejected along the same rationale. Thus, Krishnamurthy and Jamtgaard disclose every limitation of Claim 11 and provide proper reasons to combine, as indicated in the above rejections for Claim 1.

In addition, Krishnamurthy teaches:

retrieving a second data file, the second data file associated with the first data file,

(See Paragraph [0033]→ Krishnamurthy discloses this limitation, as clearly indicated in the cited text [e.g., the tree traversal is performed on the second tree guided by the selected data of the first tree (e.g., second data file associated with first data file.)

Claim 12,

Krishnamurthy and Jamtgaard teach the method of claim 11 and further comprise:

responsive to identifying non-matching content present only in the first file: adding, at least one new token to the syntax tree

(Krishnamurthy teaches retrieving the stored version of the data file with the received data file in the form of AST is being compared to identify non-matching content (new/dynamic content) using the TREEDIFF algorithm, (see Figure. 5 and page 5 Para [0077] --> Page 6 Para [0083], e.g., TreeDiff algorithm). In addition, Krishnamurthy is parsing only the TREEDIFF (the non-matching content between the VIEW (A, B, C) and PAGE2 (A, X, Y, C)) to form a subtree of the non-matching content. This is generally set forth at page 8 Para [0107-0108] and illustrated in figure 7.)

<u>In addition, Krishnamurthy does not expressly teach, but Jamtgaard teaches:</u>

A virtual browser,

(@ Figure 7 and @ Col. 10 Lines 21-47→ Jamtgaard discloses this limitation, as clearly indicated in the cited text [e.g., the content 20 is fed into ***a virtual browser*** 100 as described briefly above which passes on the content to the XML engine 46. Using the particular page-type rules sets stored in the database 47, the XML engine 46 may convert the content into relational markup language (RML) format 24. In more detail, a session on the Internet is handled by the virtual browser 100 located on the translation server 12. This virtual browser provides the important functionality of proxying javascript (*using a Javascript proxy engine 102*) and cookies (using a cookie proxy engine 104) for the target devices.)

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified Krisnamurhty's method for efficiently parsing received data file that is related to the HTML *Diff* algorithm, and utilizing a proxy and some graphical user interface (GUI) code and sending the augmented page (named page 1') to the user (named view client) (see Krisnamurhty at page 4 para[0063], to include the virtual browser as taught by Jamtgaard, because they are from the same field of endeavor of Viewing and parsing , and provides a predictable result of said content delivery may include intelligently harvesting content from a web page to provide multiple different web servers to send information to various information appliances. In particular, the system receives incoming content on-the-fly from an Internet content provider thereby allowing for dynamic information generation utilized the virtual browser; see Jamtgaard @ at column 4 lines 21-26 & @ Col. 10 Lines 21-47.

Regarding ***independent claim 13,***

Claim 13 recites a system to implement a method recited in Claim 1. Thus, Krishnamurthy and Jamtgaard disclose every limitation of Claim 15 and provide proper reasons to combine, as indicated in the above rejections for Claim 1 - Also See Krishnamurthy at Para [0011], disclose databases (cache) - Also See Krishnamurthy at Para [0063] discloses a proxy that augments a web page with some graphical user interface (GUI) code and sends the augmented page (named page 1') to the user (named view client); also at Para [0036] Krishnamurthy discloses a PAGEDIFF algorithm and at para [0109], described the use of a parser to form the Abstract Syntax Tree (AST.)

<u>In addition, Krishnamurthy do not expressly teach, but Jamtgaard teaches:</u>

a virtual browser for retrieving content from content servers;

(@ Figure 7 and @ Col. 10 Lines 21-47→ Jamtgaard discloses this limitation, as clearly indicated in the cited text [e.g., the content 20 is fed into ***a virtual browser*** 100 as described briefly above which passes on the content to the XML engine 46. Using the particular page-type rules sets stored in the database 47, the XML engine 46 may convert the content into relational markup language (RML) format 24. In more detail, a session on the Internet is handled by the virtual browser 100 located on the translation server 12. This virtual browser provides the important functionality of proxying javascript

(*using a Javascript proxy engine 102*) and cookies (using a cookie proxy engine 104) for the target devices.)

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified Krisnamurhty's method for efficiently parsing received data file that is related to the HTML *Diff* algorithm, and utilizing a proxy and some graphical user interface (GUI) code and sending the augmented page (named page 1') to the user (named view client) (see Krisnamurhty at page 4 para[0063], to include the virtual browser as taught by Jamtgaard, because they are from the same field of endeavor of Viewing and parsing , and provides a predictable result of said content delivery may include intelligently harvesting content from a web page to provide multiple different web servers to send information to various information appliances. In particular, the system receives incoming content on-the-fly from an Internet content provider thereby allowing for dynamic information generation utilized the virtual browser; see Jamtgaard @ at column 4 lines 21-26 & @ Col. 10 Lines 21-47.

Regarding ***independent claim 15,***

Claim 15 recites an intermediary to implement a method recited in Claim 1. Thus, Krishnamurthy and Jamtgaard disclose every limitation of Claim 15 and provide proper reasons to combine, as indicated in the above rejections for Claim 1 - Also See Marullo at column 4 lines 49-55, disclose a virtual browser (e.g. an intermediary).

Claims 16-18 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Krishnamurthy et al.,** (US 20060242145A1- Division of 10/177,783- filed 04/04/2002) [hereinafter "Krishnamurthy"], in view of **Basani et al.,** (US006718361B1 - filed 04/07/2000) [hereinafter "Basani"].

Regarding ***independent claim 16***, Krishnamurthy teaches:

a method for efficiently parsing received data file,

(See figure(s) 4-5 and at page 5 Paragraph [0081] →Krishnamurthy teaches this limitation, as clearly indicated in the cited text [e.g., parsing a web page (e.g. a data file) into an abstract syntax tree (AST).

comprising: identifying, by the service responsive to a rule, that the object is to be tracked; parsing, by the service responsive to identifying that the object is to be tracked, content of the received web page to create an abstract syntax tree;

(See Paragraph [0076] and [0016] →Krishnamurthy teaches tracking changes in pages by computing page differences [0076] to locate the desired clip in the new page, this is often referred to as a rule-based approach [0016]. In addition Krishnamurthy further discloses parsing a web page (e.g. a data file) into an abstract syntax tree (AST) [See Para [0081] and illustrated at figure(s) 4-5.)

storing, by the service, the abstract syntax tree and the content to the cache; determining, by the service, that the object of a second received web page is stored in the cache; **retrieving, by the service responsive to determining that the object is stored in the cache, the abstract syntax tree and the content from the cache;**

(See figure(s) 4-5 and at page 5 Paragraph [0077-0081]→ Krishnamurthy discloses a ***parser*** first transforms a web page (HTML Pages) into an ***abstract syntax tree (AST),*** wherein the different between HTML pages by comparison utilized HTML*Diff* algorithm and compare the parse tree or abstract syntax tree representations of the documents. Then the token sequences corresponding to the view and the new page are then fed into the FlatDiff stage, which computes a shortest edit sequence (a shortest edit sequence is ***THE DIFF BETWEEN Page 1 and PAGE 2*** @ Para [0067]) using flat edit sequence calculation algorithm. By locating the matching tokens in the original ASTs, the extraction stage outputs the desired clip. [0077-0081]. In addition Krishnamurthy further discloses the system polls the target page of a view periodically, it may refresh the view definition by applying the clip extraction mechanism and ***storing the fresher version of the page and its clip in place of the old view*** (e.g., dynamic caching Vs. Static), allowing the system to adapt to incremental changes smoothly instead of allowing gradual changes to accumulate beyond the system's ability to recognize them using simpler means (@ Para 157).)

comparing, by the service, the second received web page to the content to identify non-matching content in the second web page; parsing, by the service, only the non-matching content to generate a subtree;

(See Figure. 7 and 8 and @ Page 5 Para [0076] --> Krisnamurhty discloses **tracking** changes in pages by **computing page differences** (e.g., **the non-matching content**); that is only transmitting **the page difference** (non-matching) to **reconstruct the new page**. Also Krisnamurhty further discloses to compute the edit sequence between two subtrees (or trees) [which means to compute **the differences** between two documents of two trees [@ Page 5 Para [0067]); first linearized the two subtrees into two token sequences; then perform on these two token sequences **2-way FlatDiff**: to identify the sub-token sequence by locating the boundary point in each direction, then feed this pruned sub-subtree into the vanilla TreeDiff in place of the un-pruned subtree. We combine the result of the vanilla TreeDiff with the result of the FlatDiff to form the final answer. As a result of this optimization, the size of the subtrees participating in most invocations of the vanilla TreeDiff method is significantly smaller [@ figure 8 and page 9 Para [0128]]. In this case the services computing the page differences, then again parsing the DIFF to formed a sub tree. As illustrated in Fig(s) 7-8 TREE DIFF is a non-matching content of View and Page 2, then TREE DIFF is pared to form a sub tree (Clip).

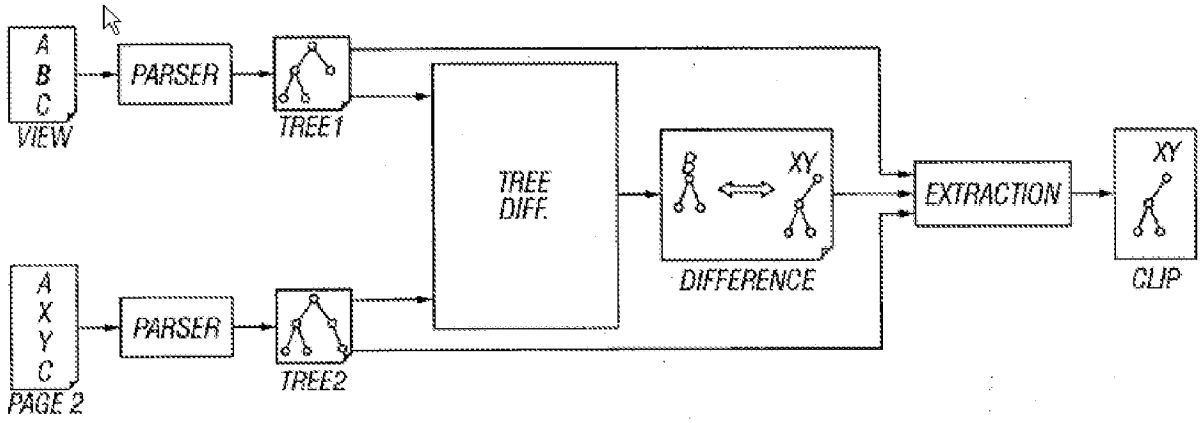

FIG. 7

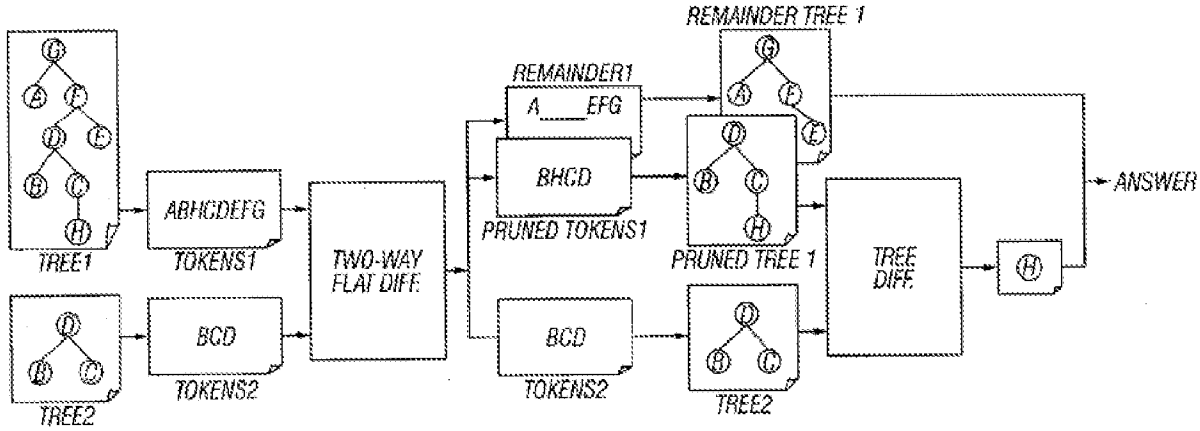

FIG. 8

This interpretation is supported by the applicant's disclosure, which is stated, “System 500 … newly retrieved version of the page and the cached copy of the page are compared using a binary **"diff" algorithm**, which identifies the **differences between the binary representation of two documents**"(See current disclosure page 11 lines 11→Line 25).

and modifying, by the service, the abstract syntax tree to comprise a token mapped to the subtree.

(See Figure. 5 and page 5 Para [0077] --> Page 6 Para [0083], and Para [0129]→ Krisnamurhty discloses the Improving TreeDiff Performance: Subtree matching wherein the difference algorithms compute the mapping from the nodes in T1 to the nodes in T2. Given such a mapping, we can identify whether the distinguished node(s) from the old page is preserved in some form inside the new page, and if so, the subtree is formed)

This interpretation is supported by the applicant's disclosure, which is stated, “System 500 … newly retrieved version of the page and the cached copy of the page are compared using a binary "diff" algorithm, which identifies the differences between the binary representation of two documents"(See current disclosure page 11 lines 11→Line 25).

In addition, Krishnamurthy do not expressly teach, but Basani teaches:

determining, by a service executing on a computing device, that a received web page comprises an object not stored in a cache;

(See Figure 1 and @ Col. 4 Lines 45-57 and @page 5 Para [0077] → Basani discloses a load-balancing processes or service-level monitors, directly determine whether a particular server has the most recent version of content. In addition Basani further discloses the CCM (Content Control Manager) will maintain a list of files currently contained in each network cache 617 on each network segment (e.g., LA, London, Paris, and Boston). When a content update occurs, the list of files contained in the

cache will be compared, and new content will be automatically distributed to the network ***cache*** (e.g., **page comprises an object not stored in a cache**) This guarantees that content being served from network caches is always up to date and fresh. The update to the cache can be a scheduled update or it can happen automatically; Mover, the CCM will contain replicated information according to a schedule or when content changes, (see Basani @ Col. 20 Lines 24-56)

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified Krisnamurhty's method for efficiently parsing received data file that is related to the HTML *Diff* algorithm, and utilizing a proxy and some graphical user interface (GUI) code and sending the augmented page (named page 1') to the user (named view client) (see Krisnamurhty at page 4 para[0063], to include a means of said determining, by a service executing on a computing device, that a received web page comprises an object not stored in a cache as taught by Basani, because they are from the same field of endeavor of efficiently parsing and caching static and dynamic web content, and provides a predictable result of said directly determine whether a particular server has the most recent version of content. This guarantee that content being served from network caches is always up to date and fresh (see Basani @ Col. 20 Lines 24-56).

Claim 17,

Krishnamurthy and Basani teach the method of claim 16 and further comprise:

wherein parsing content of the received web page to create an abstract syntax tree further comprises designating each node in the abstract syntax tree as a static node.

(See figure(s) 4-5 and at page 5 Paragraph [0081] →Krishnamurthy teaches this limitation, as clearly indicated in the cited text [e.g., improving TreeDiff performances by: reading and parsing a web page (e.g. a data file) into an abstract syntax tree (AST). The syntax tree is then linearized into a sequence of tokens, which consist of markup elements (defined by the markup language syntax and denoting structure, semantics, formatting or other information) and text strings that represent the content. Also Krisnamurhty further discloses the syntax tree includes the distinguished node(s) from the old page (e.g., static node) and newpage, wherein the old page's is preserved in some form inside the new page, and if so, the subtree is formed, see Krisnamurhty at para [0129] and illustrates in figure 7).

Claim 18,

Krishnamurthy and Basani teach the method of claim 16 and further comprise:

identifying by the token dynamic content in the abstract syntax tree.

(See figure(s) 4-5 and at page 5 Paragraph [0081] →Krishnamurthy teaches this limitation, as clearly indicated in the cited text [e.g., improving TreeDiff performances by: reading and parsing a web page (e.g. a data file) into an abstract syntax tree (AST). Also Krisnamurhty further discloses the syntax tree includes the distinguished node(s) from the old page (e.g., static node) and newpage, wherein the old page's is preserved

in some form inside the new page (e.g., token dynamic content), and if so, the subtree is formed , see Krisnamurhty at para [0129] and illustrates in figure 7).

It is noted that any citations to specific, pages, columns, lines, or figures in the prior art references and any interpretation of the references should not be considered to be limiting in any way. A reference is relevant for all it contains and may be relied upon for all that it would have reasonably suggested to one having ordinary skill in the art. See, MPEP 2123.

Response to Arguments

Applicant's arguments filed with current paper have been considered but are moot in view of the new ground(s) of rejections as cited above.

This is a NonFinal office action in order to provide applicant the opportunity to response to the new grounds of rejection, which is set forth above.

In addtion, Applicant argues that Krishnamurhty and Marullo fail to teach "*(i) comparing a stored (or cached) version of a data file (or HTML page) with a received data file (or HTML page) to identify non-matching content in the received data file (or HTML page); a*nd *(ii) parsing only the non-matching content of the received data filed (or HTML page) to form a subtree*" [see the remarks @ Page 9 bottom to top of page 10 and similar argument @ page 11, the second half and @ page 13, the first pargragh],

because the parsing in Krishnamurhty completely pares ALL the contents of the received data file. [See the remarks @ Page 10 lines 10-12], and instead of idnetified only non-matching contnet, Kishmurhty identifies Diff beween the the two three (See the remarks @ page 11 last line and page 12 first line; and similarly @ page 13 lines 13-15]

The examiner disagrees.

As indicated in the above rejection, Krishnamurthy expressly discloses that **tracking** changes in pages by **computing page differences** ;(non-matching content) that is only transmitting **the page difference** (non-matching content) to **reconstruct the new page** (sub-tree). In this case the services computing the page differences, then again parsing the DIFF (e.g. parsing only the no-matching content this time) to formed a sub tree. As illustrated in Fig(s) 7-8 TREE DIFF is a non-matching content of View and Page 2, then TREE DIFF is pared to form a sub tree (Clip).

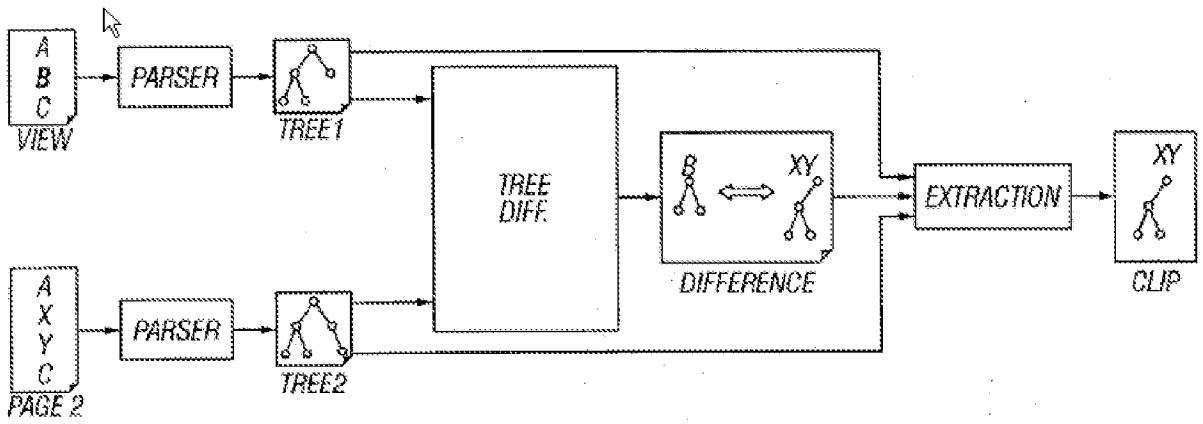

FIG. 7

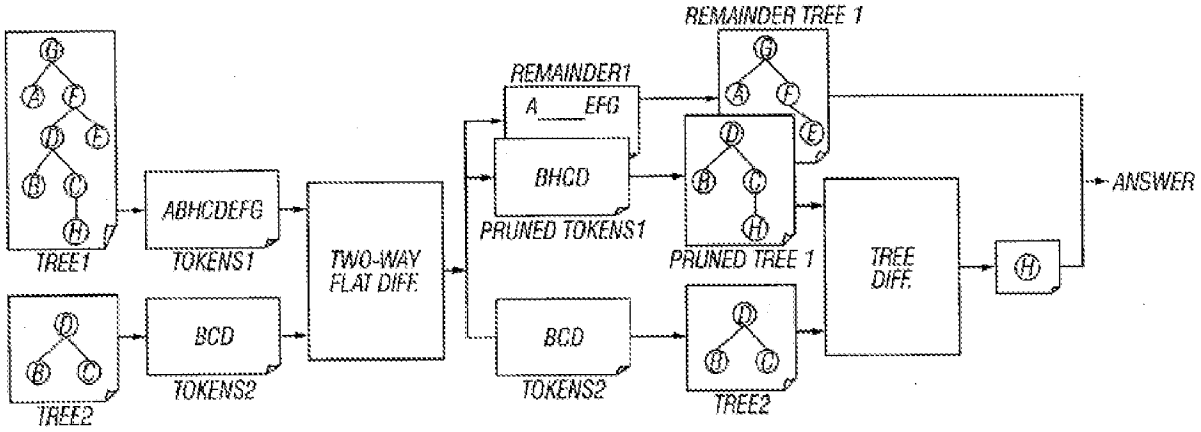

FIG. 8

Therefore, Krishnamurthy's **tracking** changes in pages by **computing page differences**; then parsing the DIFF (e.g., the non-matching) to formed a sub-tree meet the "*identify non-matching content" and " parsing only the non-matching content"* recited in the claims, whereas comparing HTML pages to "identify non-matching " in the received data file and "parsing only the non-matching content" of the received data filed to form a subtree recited in the claims.

The above interpretation is expressly supported by the instance specification, which is stated. “System 500 … newly retrieved version of the page and the cached copy of the page are compared using a binary **"diff" algorithm**, which identifies the **differences between the binary representation of two documents**"(See current disclosure page 11 lines 11→Line 25).

Accordingly, as indicated in the above rejection the Examiner respectfully maintains the rejection of claims 1, 4-5, 6, 8, 11-13, and 15-18 at this time.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Quoc A. Tran whose telephone number is 571-272-8664. The examiner can normally be reached on Mon through Fri 8AM - 5PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Doug Hutton can be reached on (571)272-4137. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Quoc A. Tran/
Examiner, Art Unit 2176

/DOUG HUTTON/
Supervisory Patent Examiner, Art Unit 2176